IN THE SPECIFICATION:

Please amend page 1, last paragraph (continuing to page 2) as follows:

In the conventional socket communication method, when a parent process 212 in the server application 211 receives a request from a process 262 in the client application 261, the parent process 212 creates a child process 213 through UNIX's fork function and the child process 213 deals with the request from the process 262. In socket communication, however, data to be communicated is once buffered in buffers inside the space of an operating system (a buffer 231 in an OS 230 and a buffer 281 in an OS 280). This has been a bottleneck in improving the throughput.

Please amend page 2, first paragraph to page 3, first full paragraph as follows:

As shown in FIG. 2, one of the endpoints (a socket 221 in a TCP/IP module 220) in socket communication is held by the process on the server (host 210) and the other (a socket 271 in a TCP/IP module 270) by the process on the client (host 260).

When the child process 213 is created by the fork on the server, the socket attributes held by the parent process are copied into the child process, which makes it possible for the parent and child processes to share and use the same socket 221. This means that the child process 213 created by the fork can communicate with the client via a network 2 as it is.

However, as shown in FIG. 3, in case of VIA, virtual interfaces (hereinafter referred to as VI) for communication lines between processes are local resources for the respective processes and even if a child process 313 in the server application 311 in a host 310 is created by the fork, it cannot share a VI 331 in a VIA module 330 for a parent process 312 to connect to a process 362 via a VI 381 in a VIA module 380 in a host 360. The parent process 312 connects to the VIA module 330 through a TCP/IP socket emulator module 320. The process 362 connects to the VIA module 380 through a TCP/IP socket emulator module 370.

Therefore, there is a problem that a child process created by the fork cannot communicate with the client because it cannot establish a new VI connection with the client in addition to a existing VI connection 50 via a network 3. As a consequence, in order to emulate the fork for socket communication in a VIA, VI connection must be established between the created child process 313 and the client application 361 in the host 360.

In some applications, there may be a case that communication is made between the parent process 312 and the process 362 in the client application 361 in the host 360 before establishment of VI connection between the child process 313 and the process 362. In this case, the child process 313 and the process 362 must take over the communication made between the parent process 312 and the process 362 without fail.

Please insert after page 22, first full paragraph the following paragraph:

First, at time a, the client issues a connection request 810 to the parent process on the server, and at time A, the parent process receives it. (In this diagram, the word "parent" in parentheses following the descriptions of various steps indicates that the step concerned takes place between the client and the parent process while the word "child" in parentheses indicates that it takes place between the client and the child process.) Then, at time B, the parent process issues connection permission 811 to the client and at time b, the client receives it. Then, at time E, the parent process issues a reconnection request 812 to the client and at time e, the client receives it. At time f, the client issues a connection request 813 to the child process and at time F, the child process receives it. Then, at time G, the child process issues connection permission 814 to the client and at time g, the client receives it. Lastly, at time h, the client issues a report of completion of reconnection 815 to the parent process, and at time H, the parent process receives it and thus a series of VI reconnection steps are completed.